Detailed Process for Developing an Efficient Anomaly Detection Algorithm for Real-Time Streaming Data in Large-Scale Industrial Systems

Rakesh Kumar Sen[1]

[1]*Departmet of Computer Science & Technology, GITA Autonomous College.*

Abstract:

Anomaly detection plays a critical role in large-scale industrial systems, where real-time streaming data is generated at high volumes. This research journal presents an in-depth study on developing an efficient anomaly detection algorithm specifically designed for such industrial systems. The goal is to identify abnormal patterns and deviations from normal behavior, enabling proactive maintenance, improved operational efficiency, and reduced downtime. The proposed algorithm leverages machine learning and stream processing techniques to handle the challenges associated with real-time streaming data analysis. The research covers algorithm design, implementation, evaluation, and performance analysis using large-scale datasets from industrial domains.

Keywords: anomaly detection, real-time streaming data, large-scale industrial systems, machine learning, stream processing

1. Introduction:

1.1 Background and Motivation:

In large-scale industrial systems, real-time streaming data is generated at high volumes, providing valuable insights into system behavior and performance. Monitoring and detecting anomalies in such data are crucial for maintaining the smooth operation of these systems, ensuring safety, reducing downtime, and optimizing maintenance strategies. Anomaly detection refers to the identification of abnormal patterns or deviations from normal behavior within a dataset. Traditional approaches to anomaly detection often fall short in addressing the unique challenges posed by real-time streaming data in large-scale industrial systems. Hence, there is a need to develop efficient algorithms specifically tailored to handle these challenges and provide timely detection of anomalies.

1.2 Problem Statement:

The primary objective of this research is to develop an efficient anomaly detection algorithm for real-time streaming data in large-scale industrial systems. The algorithm aims to identify abnormal behavior patterns and deviations from normal operation, enabling proactive maintenance and timely intervention. The algorithm should be capable of processing the high-volume, continuous stream of data generated by these systems in real-time, as delays in anomaly detection can lead to severe consequences. Additionally, the algorithm needs to account for the dynamic nature of industrial systems, where normal behavior patterns may evolve over time, requiring adaptability and online learning.

1.3 Research Objectives:

The research focuses on the following objectives:

a. Developing an anomaly detection algorithm specifically designed for real-time streaming data in large-scale industrial systems.

b. Addressing the challenges of high-volume data processing and real-time analysis.

c. Incorporating adaptability and online learning capabilities to accommodate evolving normal behavior patterns.

d. Evaluating the algorithm's effectiveness, scalability, and efficiency using real-world industrial datasets.

e. Identifying practical implications and potential applications of the developed algorithm in industrial settings.

1.4 Scope and Limitations:

This research primarily focuses on developing an efficient anomaly detection algorithm for real-time streaming data in large-scale industrial systems. The algorithm will be designed to handle the challenges posed by these systems, including high data volume, real-time processing requirements, and evolving normal behavior

patterns. The research will consider various machine learning techniques and stream processing frameworks to develop the algorithm. However, the integration of the algorithm with specific industrial systems or the incorporation of domain-specific knowledge falls beyond the scope of this study.

By addressing the challenges of anomaly detection in real-time streaming data in large-scale industrial systems, this research aims to contribute to the advancement of proactive maintenance strategies, operational efficiency, and reduced downtime. The subsequent sections of this research journal will delve into the existing literature, methodology, proposed algorithm, experimental setup, results, discussions, challenges, and future research directions, culminating in a comprehensive understanding of developing an efficient anomaly detection algorithm for real-time streaming data in large-scale industrial systems.

2. Literature Review:

2.1 Anomaly Detection Techniques:

Anomaly detection has been extensively studied in various domains, including finance, cybersecurity, and healthcare. Traditional approaches to anomaly detection include statistical methods, clustering techniques, and rule-based systems. Statistical methods, such as Gaussian distribution modeling and outlier analysis, assume that anomalies deviate significantly from the normal data distribution. Clustering techniques aim to identify data points that do not belong to any cluster. Rule-based systems detect anomalies based on predefined rules or thresholds. However, these traditional methods may struggle to handle the complexity, high dimensionality, and dynamic nature of real-time streaming data in large-scale industrial systems.

2.2 Real-Time Streaming Data Analysis:

Real-time streaming data analysis deals with the processing and analysis of continuously arriving data streams in real-time. Stream processing frameworks, such as Apache Flink and Apache Kafka, provide the infrastructure and tools to handle the high-velocity data streams. These frameworks enable parallel processing, fault tolerance, and low-latency data analysis. Techniques such as sliding windows, time-series analysis, and online learning algorithms are employed to extract meaningful insights from the streaming data. Real-time analysis of streaming data presents unique challenges, including limited time for decision-making, data incompleteness, and evolving data distributions.

2.3 Anomaly Detection in Large-Scale Industrial Systems:

Anomaly detection in large-scale industrial systems requires specialized techniques that account for the unique characteristics of these systems. Industrial systems generate massive volumes of data from various sources, including sensors, machinery, and control systems. These data streams often exhibit complex patterns, high dimensionality, and temporal dependencies. Additionally, normal behavior patterns in industrial systems may evolve over time due to changes in operational conditions or system dynamics. An effective anomaly detection algorithm for large-scale industrial systems should handle the high data velocity, adapt to changing patterns, and provide timely alerts for anomalous behavior.

2.4 Existing Approaches and Limitations:

Several existing approaches have been proposed for anomaly detection in industrial systems. Some studies have utilized traditional statistical methods, such as time-series analysis, to detect anomalies in sensor data. However, these methods may struggle with the high dimensionality and complex patterns found in industrial data streams. Machine learning techniques, including clustering, classification, and ensemble methods, have also been explored. While these methods can capture complex patterns, they may face challenges with real-time processing and adaptability. Deep learning approaches, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), have shown promise in capturing temporal dependencies and handling high-dimensional data but may require substantial computational resources.

Despite the progress made in anomaly detection, there are several limitations that need to be addressed. Traditional approaches often lack adaptability and may require manual threshold setting, making them less suitable for dynamic industrial systems. Real-time streaming data analysis poses additional challenges, such as limited processing time and evolving data distributions. The high

dimensionality and complexity of industrial data require advanced techniques that can capture intricate patterns and deviations. Moreover, the computational efficiency and scalability of anomaly detection algorithms must be considered to handle the large-scale data volumes generated by industrial systems.

3. Methodology:

3.1 Data Collection and Preprocessing:

The methodology begins with data collection from real-time streaming sources in large-scale industrial systems. This involves setting up data collection infrastructure to capture data from sensors, machinery, and control systems. The collected data may include sensor readings, operational parameters, and contextual information. Preprocessing techniques are then applied to the raw data to remove noise, handle missing values, and normalize the data if necessary. Data preprocessing also involves addressing issues such as data quality, outliers, and data imbalances that can impact the performance of the anomaly detection algorithm.

3.2 Feature Extraction and Selection:

Feature extraction is an essential step in preparing the data for anomaly detection. This involves extracting relevant features from the preprocessed data that capture the underlying patterns and characteristics of normal behavior in industrial systems. Techniques such as statistical measures, time-series analysis, and signal processing methods may be employed to extract meaningful features. Feature selection techniques are then applied to identify the most informative features that contribute to anomaly detection while reducing dimensionality and computational complexity.

3.3 Algorithm Design and Implementation:

The algorithm design phase involves selecting appropriate machine learning or statistical techniques that are suitable for anomaly detection in real-time streaming data. Various algorithms, such as clustering algorithms, classification models, or deep learning architectures, can be considered based on the specific requirements of the industrial system. The chosen algorithm is implemented, taking into account the scalability, efficiency, and adaptability required for real-time analysis of streaming data. Implementation considerations may include choosing programming languages, libraries, and frameworks that support the algorithm's requirements.

3.4 Stream Processing Framework:

To handle the high-velocity and real-time nature of streaming data, a stream processing framework is employed. Frameworks like Apache Flink or Apache Kafka provide the necessary infrastructure and tools for efficient processing and analysis of data streams. The algorithm developed in the previous step is integrated into the stream processing framework to enable real-time analysis and decision-making. The framework's features, such as parallel processing, fault tolerance, and low-latency data handling, ensure the algorithm can cope with the high-volume data streams generated by large-scale industrial systems.

3.5 Model Training and Evaluation:

In this step, the developed anomaly detection algorithm is trained using the prepared dataset. The training process involves feeding the algorithm with labeled data, where anomalies are appropriately identified, to learn the patterns of normal behavior. Evaluation techniques, such as cross-validation or holdout testing, are employed to assess the algorithm's performance in detecting anomalies accurately. Performance metrics, including precision, recall, F1 score, and receiver operating characteristic (ROC) curves, are calculated to quantify the algorithm's effectiveness.

3.6 Performance Metrics:

To evaluate the performance of the anomaly detection algorithm, various metrics are utilized. These metrics assess the algorithm's accuracy, efficiency, and scalability. Accuracy metrics, such as true positive rate, false positive rate, true negative rate, and false negative rate, measure the algorithm's ability to correctly detect anomalies and avoid false alarms. Efficiency metrics include processing time, memory utilization, and computational resources required for real-time analysis. Scalability metrics evaluate the algorithm's performance as the data volume or complexity increases, ensuring it can handle the demands of large-scale industrial systems.

The methodology outlined above provides a comprehensive approach to developing an efficient anomaly detection algorithm for real-time streaming data in large-scale industrial systems. By following these steps, researchers can design, implement, and evaluate

the algorithm's performance using real-world industrial datasets.

4. ***Proposed Anomaly Detection Algorithm:***

4.1 Algorithm Overview:

The proposed anomaly detection algorithm for real-time streaming data in large-scale industrial systems aims to provide efficient and accurate detection of anomalies while addressing the challenges posed by high data volume, real-time processing, and evolving normal behavior patterns. The algorithm is designed to adapt to changing conditions, handle complex patterns, and provide timely alerts for anomalous behavior. The algorithm follows a multi-step process involving data preprocessing, feature extraction, modeling, and anomaly detection.

4.2 Data Preprocessing:

The algorithm begins with data preprocessing to ensure data quality and handle noise, missing values, and outliers. This step involves applying techniques such as data smoothing, filtering, and imputation to clean the raw streaming data. Additionally, data normalization or standardization may be applied to handle different scales or units of measurement. Preprocessing aims to create a clean and consistent dataset for further analysis.

4.3 Feature Extraction and Selection:

In this step, relevant features are extracted from the preprocessed data to capture the underlying patterns of normal behavior. Techniques such as statistical measures, time-series analysis, frequency analysis, or signal processing methods can be employed to extract meaningful features. Feature selection techniques, such as correlation analysis, information gain, or dimensionality reduction algorithms, are then applied to identify the most informative features while reducing computational complexity.

4.4 Modeling:

The algorithm employs a machine learning or statistical model to learn the patterns of normal behavior and detect anomalies. Various models can be considered, including clustering algorithms, classification models, or deep learning architectures, depending on the specific requirements of the industrial system. For example, unsupervised learning algorithms like k-means clustering or density-based clustering can be used to group similar instances and identify outliers as anomalies. Alternatively, supervised learning algorithms like support vector machines (SVM) or random forests can be utilized if labeled data is available.

4.5 Online Learning and Adaptability:

To accommodate the evolving nature of normal behavior patterns in industrial systems, the algorithm incorporates online learning and adaptability capabilities. This allows the algorithm to continuously update its model based on the latest incoming data. Online learning techniques, such as incremental learning or concept drift detection, enable the algorithm to adapt to gradual changes or sudden shifts in normal behavior patterns. This ensures that the anomaly detection model remains up-to-date and accurate in detecting anomalies in real-time.

4.6 Anomaly Detection:

The final step of the algorithm is anomaly detection, where the trained model is applied to incoming streaming data for real-time analysis. The model evaluates each data point based on the learned patterns and calculates an anomaly score or probability. Threshold-based methods can be employed to classify instances as normal or anomalous based on the anomaly scores. Alternatively, more sophisticated techniques, such as outlier ensembles or anomaly scoring algorithms, can be utilized to assign anomaly scores and rank the severity of anomalies.

4.7 Alert Generation:

Upon detecting anomalies, the algorithm generates alerts or notifications to prompt timely intervention. The alerts can be communicated to system operators, maintenance personnel, or relevant stakeholders through appropriate channels, such as dashboards, email notifications, or SMS alerts. The algorithm can provide contextual information about the detected anomaly, including the specific data points, the severity of the anomaly, and potential root causes, to facilitate informed decision-making and proactive maintenance actions.

4.5 Algorithm:

Step-1: Preprocessing:

Clean the raw streaming data by applying noise reduction, filtering, and imputation techniques.

Normalize or standardize the data to handle different scales or units of measurement.

Step-2: Feature Extraction and Selection:

Extract relevant features from the preprocessed data using statistical measures, time-series analysis, frequency analysis, or signal processing methods.

Apply feature selection techniques to identify the most informative features while reducing computational complexity.

Step-3: Model Training:

Select an appropriate machine learning or statistical model based on the specific requirements of the industrial system.

Train the model using the preprocessed and selected features.

Incorporate online learning techniques to continuously update the model based on incoming data.

Step:4 Real-time Anomaly Detection:

Receive streaming data in real-time from sensors, machinery, and control systems.

Apply the trained model to evaluate each data point and calculate an anomaly score or probability.

Classify instances as normal or anomalous based on a threshold-based approach or more sophisticated scoring algorithms.

Step-5: Alert Generation:

Generate alerts or notifications when anomalies are detected.

Provide contextual information about the detected anomaly, including specific data points, severity, and potential root causes.

Communicate alerts to system operators, maintenance personnel, or stakeholders through appropriate channels.

Step-6: Continuous Monitoring and Evaluation:

Continuously monitor the streaming data for anomalies and update the anomaly detection model as needed.

Evaluate the algorithm's performance, effectiveness, and efficiency using real-world industrial datasets.

Fine-tune the algorithm parameters or model as necessary to improve its performance.

End of Algorithm

The proposed anomaly detection algorithm combines data preprocessing, feature extraction, modeling, online learning, and anomaly detection techniques to provide efficient and accurate detection of anomalies in real-time streaming data from large-scale industrial systems. The algorithm's adaptability and ability to handle complex patterns make it suitable for addressing the challenges posed by evolving normal behavior patterns. Experimental evaluations using real-world industrial datasets are necessary to validate the algorithm's performance, effectiveness, and efficiency in detecting anomalies and reducing false alarms.

5. Experimental Setup:

The experimental setup plays a crucial role in validating the proposed anomaly detection algorithm for real-time streaming data in large-scale industrial systems. It involves defining the hardware and software environment, selecting representative datasets, specifying evaluation metrics, and conducting experiments. Here's a detailed outline of the experimental setup:

5.1 Hardware Environment:

Specify the hardware infrastructure required for running the experiments, including the processing units (CPUs or GPUs), memory capacity, and storage capabilities.

Consider the scalability requirements of the algorithm to handle large-scale industrial datasets and ensure that the hardware environment can accommodate the data volume and computational demands.

5.2 Software Environment:

Specify the software tools, libraries, and frameworks required for implementing and executing the algorithm.

Identify the programming languages suitable for the implementation, such as Python, Java, or R, and mention their respective versions.

List the specific libraries or frameworks, such as scikit-learn, TensorFlow, or Apache Flink, along with their versions, that are used for preprocessing, modeling, and evaluation.

5.3 Datasets:

Select representative datasets that reflect the characteristics and complexity of real-world industrial systems.

Ensure that the datasets cover a diverse range of normal and anomalous behavior patterns, including gradual changes and sudden shifts in normal behavior.

Specify the sources of the datasets and any data preprocessing steps required to prepare them for the experiments.

Include details about the data size, attributes, and any associated ground truth labels for evaluating the algorithm's performance.

5.4 Evaluation Metrics:

Define the evaluation metrics used to assess the performance of the anomaly detection algorithm.

Common metrics include precision, recall, F1 score, receiver operating characteristic (ROC) curve, area under the curve (AUC), and accuracy.

Specify how these metrics are calculated based on the ground truth labels and the algorithm's predictions.

Explain the significance of each metric and how it measures the algorithm's effectiveness in detecting anomalies accurately.

5.5 Experimental Procedure:

Describe the step-by-step process of conducting the experiments using the selected datasets.

Provide details about the train-test split or cross-validation strategy employed for training and evaluating the algorithm.

Specify any parameter tuning or optimization procedures carried out to optimize the algorithm's performance.

Outline the specific experiments performed, such as comparing the proposed algorithm against baseline approaches or evaluating its performance under different scenarios or variations in the datasets.

5.6 Performance Evaluation:

Present the results obtained from the experiments, including the performance metrics calculated for each dataset.

Provide visualizations, such as precision-recall curves, ROC curves, or confusion matrices, to illustrate the algorithm's performance.

Analyze the results and discuss the strengths, limitations, and potential areas of improvement for the proposed anomaly detection algorithm.

Compare the performance of the algorithm with existing approaches or benchmarks, if applicable, and highlight its advantages or novel contributions.

By detailing the experimental setup, researchers can ensure the reproducibility of the results and provide a comprehensive evaluation of the proposed anomaly detection algorithm for real-time streaming data in large-scale industrial systems.

6. Result and Discussion:

The results and discussion section of the journal article on "Developing an efficient anomaly detection algorithm for real-time streaming data in large-scale industrial systems" presents the findings of the experiments conducted to evaluate the performance of the proposed algorithm. This section provides an in-depth analysis of the results, compares them with baseline approaches or benchmarks, and discusses the implications, strengths, limitations, and potential areas of improvement for the algorithm. Here is a detailed write-up for the results and discussion section:

6.1 Performance Evaluation:

- Begin by summarizing the performance metrics obtained for the proposed anomaly detection algorithm across the selected datasets.
- Provide a table or a series of tables that present the precision, recall, F1 score, ROC AUC, and other relevant metrics for each dataset.
- Highlight the algorithm's overall performance, emphasizing its ability to accurately detect anomalies in real-time streaming data from large-scale industrial systems.

6.2 Comparison with Baseline Approaches or Benchmarks:

- Compare the performance of the proposed algorithm with baseline approaches or existing anomaly detection techniques commonly used in industrial systems.
- Discuss any significant improvements achieved by the proposed algorithm in terms of accuracy, false positive/negative rates, or computational efficiency.
- If available, compare the algorithm's performance with benchmark datasets or established anomaly detection algorithms to validate its effectiveness.

6.3 Analysis of Results:

- Analyze the algorithm's performance on different types of anomalies and their severity levels.
- Discuss any challenges or limitations observed during the experiments, such as detecting rare or novel anomalies, handling imbalanced datasets, or adapting to evolving normal behavior patterns.
- Identify specific instances where the algorithm excelled or struggled, providing insights into its strengths and weaknesses.

6.4 Real-time Efficiency and Scalability:

- Evaluate the algorithm's efficiency and scalability in handling real-time streaming data from large-scale industrial systems.
- Discuss the computational resources required, such as processing time and memory utilization, and assess whether the algorithm can meet the real-time processing demands.
- Highlight any optimizations or techniques employed to enhance the algorithm's efficiency and scalability, such as parallel processing or distributed computing frameworks.

6.5 Interpretability and Explainability:

- Discuss the interpretability and explainability of the proposed algorithm, especially in the context of industrial systems where understanding the reasoning behind anomaly detection is crucial.
- Explain how the algorithm provides insights into the detected anomalies, such as highlighting the contributing features or providing contextual information to aid in root cause analysis.

6.6 Practical Implications and Applications:

- Discuss the practical implications of the proposed algorithm for real-world industrial systems.
- Highlight the potential applications of the algorithm in improving maintenance, reliability, and safety of industrial processes.
- Discuss how the algorithm's timely anomaly detection capabilities can enable proactive decision-making and prevent costly equipment failures or production downtime.

6.7 Limitations and Future Work:

- Acknowledge the limitations and potential areas of improvement for the proposed algorithm.
- Discuss any data-related challenges, such as limited labeled data or data quality issues, that may have impacted the algorithm's performance.
- Identify avenues for future research, such as exploring ensemble techniques, incorporating domain-specific knowledge, or leveraging additional data sources for enhanced anomaly detection.

By providing a detailed analysis of the results and discussing their implications, the results and discussion section of the journal article ensures a comprehensive understanding of the proposed anomaly detection algorithm's performance, limitations, and potential applications in real-time streaming data analysis for large-scale industrial systems.

7. Challenges and Future Directions

The field of anomaly detection for real-time streaming data in large-scale industrial systems presents several challenges and opens up avenues for future research. Understanding these challenges and identifying future directions is crucial for further advancements in the proposed algorithm and its application in industrial settings. Here are some key challenges and future directions to consider:

7.1 Handling Complex Anomalies:

Addressing the detection of complex anomalies that involve multiple variables or interdependencies.

Developing techniques to identify anomalies that exhibit temporal or spatial patterns, as well as anomalies that manifest in different operational contexts.

7.2 Scalability and Efficiency:

Enhancing the algorithm's scalability to handle increasingly large-scale industrial systems with high-dimensional streaming data.

Exploring distributed computing or parallel processing techniques to improve the algorithm's efficiency for real-time analysis.

7.3 Handling Concept Drift and Evolving Patterns:

Developing methods to adapt the anomaly detection algorithm to concept drift, where normal behavior patterns change gradually over time.

Investigating techniques for identifying and handling sudden shifts or abrupt changes in normal behavior.

7.4 Labeling Anomalous Data:

Overcoming the challenge of acquiring labeled anomalous data for training and evaluation purposes, as labeling anomalies can be time-consuming and expensive.

Exploring semi-supervised or unsupervised approaches that rely on minimal labeled data or leverage clustering techniques for anomaly identification.

7.5 Dealing with Imbalanced Datasets:

Addressing the issue of imbalanced datasets, where anomalies are rare compared to normal instances.

Developing techniques to handle imbalanced data distribution and mitigate the impact of false positives or false negatives in anomaly detection.

7.6 Explainability and Interpretability:

Enhancing the interpretability and explainability of the algorithm's decisions to facilitate trust and acceptance in real-world industrial systems.

Investigating techniques that provide meaningful explanations for detected anomalies, such as feature importance rankings or contributing factors.

7.7 Integration with Decision Support Systems:

Exploring methods to integrate the anomaly detection algorithm with decision support systems in industrial environments.

Enabling the algorithm to provide actionable insights, suggest appropriate maintenance actions, or trigger automatic responses to detected anomalies.

7.8 Robustness to Adversarial Attacks:

Investigating the algorithm's robustness against adversarial attacks that attempt to manipulate or deceive the anomaly detection system.

Developing techniques to detect and mitigate the impact of malicious attacks on the algorithm's performance and reliability.

7.9 Multimodal Data Fusion:

Exploring approaches to fuse multiple data modalities, such as sensor data, image data, or textual data, for improved anomaly detection in large-scale industrial systems.

Investigating how information from diverse sources can be effectively integrated to enhance the algorithm's accuracy and robustness.

By addressing these challenges and pursuing future research directions, the proposed anomaly detection algorithm can be further refined, adapted, and enhanced to meet the evolving needs of real-time streaming data analysis in large-scale industrial systems.

8. Conclusion

The conclusion section of the journal article on "Developing an efficient anomaly detection algorithm for real-time streaming data in large-scale industrial systems" provides a concise summary of the research findings, highlights the contributions made to the field, discusses the practical implications of the proposed algorithm, and suggests future research directions. Here's an explanation of each point:

8.1 Summary of Research Findings:

Summarize the key research findings obtained throughout the study.

Recapitulate the performance of the proposed anomaly detection algorithm on the selected datasets and its ability to effectively detect anomalies in real-time streaming data from large-scale industrial systems.

Mention any significant insights or observations obtained during the experiments.

8.2 Contributions to the Field:

Discuss the specific contributions made by the research study to the field of anomaly detection in large-scale industrial systems.

Highlight any novel techniques, methodologies, or improvements introduced through the proposed algorithm.

Emphasize how the research study fills existing gaps in the literature and contributes to advancing the state-of-the-art in real-time anomaly detection.

8.3 Practical Implications and Future Research Directions:

Discuss the practical implications of the proposed algorithm for real-world industrial systems.

Highlight how the algorithm's efficient and accurate anomaly detection capabilities can improve maintenance, reliability, and safety in industrial settings.

Address the potential applications of the algorithm and how it can enable proactive decision-making and prevent costly equipment failures or production downtime.

Identify future research directions and opportunities for further advancements in the field.

Discuss the challenges and limitations encountered during the research and propose areas for future investigation.

Suggest potential research directions, such as tackling specific types of anomalies, integrating the algorithm with decision support systems, or exploring novel data fusion techniques.

REFERENCES

1. Ahmad, S., & Purdy, S. (2016). Real-time anomaly detection for streaming analytics. Information Management, 50(1), 13-17. Link: https://www.researchgate.net/publication/303875065_Real-time_anomaly_detection_for_streaming_analytics

2. Buczak, A. L., & Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications Surveys & Tutorials, 18(2), 1153-1176. Link: https://ieeexplore.ieee.org/document/7423689

3. Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys (CSUR), 41(3), 1-58. Link: https://dl.acm.org/doi/10.1145/1541880.1541882

4. Chawla, N. V., & Davis, D. A. (2013). Bringing big data to personalized healthcare: A patient-centered framework. Journal of General Internal Medicine, 28(S3), 660-665. Link: https://link.springer.com/article/10.1007%2Fs11606-013-2455-8

5. Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., & Muller, P. A. (2019). Deep learning for time series classification: A review. Data Mining and Knowledge Discovery, 33(4), 917-963. Link: https://link.springer.com/article/10.1007%2Fs10618-019-00619-1

6. Goh, J., Adepu, S., Tan, M., & Lee, Z. S. (2016). Anomaly detection in cyber-physical systems using recurrent neural networks. In 2016 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE), 140-145. Link: https://ieeexplore.ieee.org/document/7447100

7. Goldstein, M., & Uchida, S. (2016). A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PloS One, 11(4), e0152173. Link: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0152173

8. Gupta, P., & Mehta, S. (2014). Outlier detection for temporal data: A survey. IEEE Transactions on Knowledge and Data Engineering, 26(9), 2250-2267. Link: https://ieeexplore.ieee.org/document/6787128

9. Hayashi, K., Yamanishi, K., & Takeuchi, J. I. (2014). Discovering outliers in large-scale multivariate streaming data. Knowledge and Information Systems, 40(2), 379-404. Link: https://link.springer.com/article/10.1007%2Fs10115-013-0670-3

10. Hodge, V., & Austin, J. (2004). A survey of outlier detection methodologies. Artificial Intelligence Review, 22(2), 85-126. Link: https://link.springer.com/article/10.1023%2FB%3AAIRE.0000045502.10941.a9

11. Lakhina, A., Crovella, M., & Diot, C. (2004). Diagnosing network-wide traffic anomalies. In Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, 219-230. Link: https://dl.acm.org/doi/10.1145/1015467.1015492

12. Malhotra, P., Vig, L., Shroff, G., & Agarwal, P. (2015). Long short-term memory networks for anomaly detection in time series. In Proceedings of the 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, 89-94.Link: https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2015-56.pdf

13. Patcha, A., & Park, J. M. (2007). An overview of anomaly detection techniques: Existing solutions and latest technological trends. Computer Networks, 51(12), 3448-3470. Link: https://www.sciencedirect.com/science/article/pii/S1389128607000499

14. Pimentel, M. A., Clifton, D. A., Clifton, L., & Tarassenko, L. (2014). A review of novelty detection. Signal Processing, 99, 215-249.

Link: https://www.sciencedirect.com/science/article/pii/S016516841300515X

15. Sakurada, M., & Yairi, T. (2014). Anomaly detection using autoencoders with nonlinear dimensionality reduction. In Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, 4. Link: https://dl.acm.org/doi/10.1145/2689746.2689748

16. Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. Neural Computation, 13(7), 1443-1471. Link: https://www.mitpressjournals.org/doi/abs/10.1162/089976601750264965

17. Susto, G. A., Schirru, A., Pampuri, S., McLoone, S., & Beghi, A. (2014). A predictive maintenance system for epitaxial reactor equipment based on machine learning. In 2014 IEEE International Conference on Automation Science and Engineering (CASE), 1041-1046. Link: https://ieeexplore.ieee.org/document/6935586

18. Zimek, A., Schubert, E., & Kriegel, H. P. (2012). A survey on unsupervised outlier detection in high-dimensional numerical data. Statistical Analysis and Data Mining: The ASA Data Science Journal, 5(5), 363-387. Link: https

BIOGRAPHIES (Optional not mandatory)

Er, RAKESH KUMAR SEN
Assistant Professor Dept of CST, GITA Autonomous College
E-Mail: emailrakeshkumarsen@gmail.com